

Lab 8. Condition Variables

XUE Jin

jinxue@cse.cuhk.edu.hk

Lab Eight

- ▣ In this lab, you will learn how to use condition variables in pthread library to implement wakeup-enabled multi-threading programs. In particular, we will use the following functions:
 - ◆ pthread_cond_init()
 - ◆ pthread_cond_wait()
 - ◆ pthread_cond_signal()

Condition Variable

- All conds must be properly initialized.
 - ◆ One way: using `PTHREAD_COND_INITIALIZER`

```
pthread_cond_t cond = PTHREAD_COND_INITIALIZER;
```

- ◆ The dynamic way: using `pthread_cond_init()`

```
int rc = pthread_cond_init(&cond, NULL);  
assert(rc == 0); // always check success!
```

Condition Variable

□ Provide conditional access to a critical section

◆ Interface

```
int  
pthread_cond_wait(pthread_cond_t *cv, pthread_mutex_t *mutex);  
int pthread_cond_signal(pthread_cond_t *cv);
```

◆ Usage (*main thread*)

```
pthread_mutex_lock(&m);  
if (x < 8) // or whatever else conditions are  
    pthread_cond_wait(&c, &m);  
pthread_mutex_unlock(&m);
```

◆ Usage (*child thread*)

```
pthread_mutex_lock(&m);  
x = x + 1; // or whatever your critical section is  
pthread_cond_signal(&c);  
pthread_mutex_unlock(&m);
```

Example

```
int done = 0;

void thr_exit() {
    pthread_mutex_lock(&m);
    done = 1;
    pthread_cond_signal(&c);
    pthread_mutex_unlock(&m);
}

void thr_join() {
    pthread_mutex_lock(&m);
    while (done == 0) // prevent spurious wakeup
        pthread_cond_wait(&c, &m);
    pthread_mutex_unlock(&m);
}
```

Example

```
void* child (void* arg) {
    printf("child\n");
    thr_exit();
    return NULL;
}

int main(int argc, char* argv[]) {
    printf("parent: begin\n");
    pthread_t p;
    pthread_create(&p, NULL, child, NULL);

    thr_join();
    printf("parent: end\n");
    return 0;
}
```

Q1: pthread_cond_wait(&c, &m)

- ❑ Why a mutex (m) will be passed into this function?
- ❑ Let's see what has been done inside:

```
pthread_cond_wait(pthread_cond_t* C, pthread_mutex_t* M)
{
    put this thread into wakeup queue of condtion var C.
    pthread_mutex_unlock(M);
    sleep();
    pthread_mutex_lock(M);
    take this thread out of wakeup queue of condition var C.
}
```

Q2: while (done == 0)

- ❑ Why don't we use if (done == 0) ?
- ❑ **Spurious** wakeup !
- ❑ According to POSIX description, pthread_cond_signal() **may** wakeup multiple threads in multi-core system.
- ❑ Only **target thread** can be woken, others need to **sleep again**.
- ❑ Use while (done == 0) to prevent **spurious** wakeup.

Exercise

- ▣ You are required to implement two functions like `pthread_join()` and `pthread_exit()`
- ▣ Different from the example, your implementation need to work in the case where there are more than one child thread